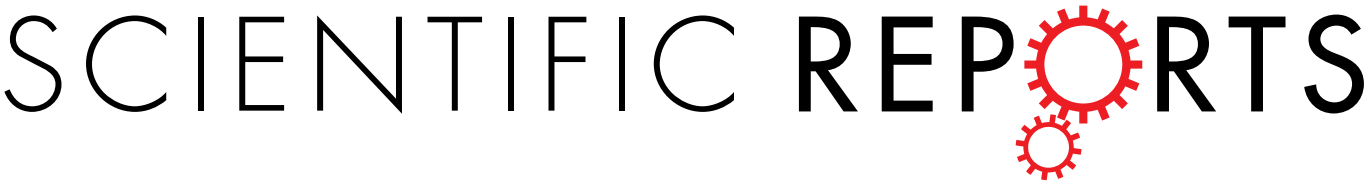

OPEN

Fragmenting networks by targeting collective influencers at a mesoscopic level

Received: 25 August 2016
Accepted: 01 November 2016
Published: 25 November 2016

Teruyoshi Kobayashi[1] & Naoki Masuda[2]

A practical approach to protecting networks against epidemic processes such as spreading of infectious diseases, malware, and harmful viral information is to remove some influential nodes beforehand to fragment the network into small components. Because determining the optimal order to remove nodes is a computationally hard problem, various approximate algorithms have been proposed to efficiently fragment networks by sequential node removal. Morone and Makse proposed an algorithm employing the non-backtracking matrix of given networks, which outperforms various existing algorithms. In fact, many empirical networks have community structure, compromising the assumption of local tree-like structure on which the original algorithm is based. We develop an immunization algorithm by synergistically combining the Morone-Makse algorithm and coarse graining of the network in which we regard a community as a supernode. In this way, we aim to identify nodes that connect different communities at a reasonable computational cost. The proposed algorithm works more efficiently than the Morone-Makse and other algorithms on networks with community structure.

Identification of influential nodes in a network is a topic of interest in network analysis, enjoying numerous applications. For example, a removal or immunization of an influential node may suppress spreading of an infectious disease that may occur later. A viral information spreading campaign starting from an influential node may be more successful than a campaign starting from other nodes. There are various notions of influential nodes, as evinced by a multitude of definitions of node's centrality corresponding to the aforementioned and other applications[1]. Among them, a major criterion of the influential node is that the removal of a node, or immunization, efficiently fragments the network into small pieces. Because the problem of finding the minimal set of nodes to be immunized to fragment the network is NP-hard[2], various immunization algorithms to determine the order of the nodes to be removed to realize efficient fragmentation of the network have been proposed[2–12], sometimes with the constraint that the information about the network is only partially available[6,13–17]. Notably, although immunizing hubs (i.e., nodes with a large degree) first is intuitive and much better than randomly selecting nodes to be immunized[18–20], many immunization algorithms outperform the hub-first immunization algorithm.

Morone and Makse proposed a scalable and powerful algorithm to sequentially remove nodes and fragment the network into small components as early as possible[9]. Founded on the message passing approach and theory of non-backtracking matrices, the method calculates the so-called collective influence (CI) for each node to rank the nodes for prioritization. Their method, which is referred to as the CI algorithm, outperforms various other known methods in model and empirical networks. In the present study, we propose a new CI-based immunization algorithm that is designed to perform well when the network has community structure.

The CI algorithm assumes that the given network is locally tree-like. In fact, a majority of empirical networks are not locally tree-like. At a microscopic level, empirical networks are usually clustered, i.e., full of triangles[1]. At a mesoscopic level, many networks are composed of communities such that links are dense within communities and sparse across different communities[21]. Although the CI algorithm also seems to work efficiently in loopy networks unless loops are not excessive[9], the performance of the CI algorithm on networks with community structure is unclear. Some extant immunization algorithms are explicitly or implicitly informed by community structure[4–6,10,15,16,22,23]. The immunization algorithms using the betweenness centrality are effective on networks with community structure[6,7,16,22,23]. However, they are not scalable due to a high computational cost of calculating the betweenness centrality[24]. For other immunization algorithms exploiting community structure of networks,

[1]Graduate School of Economics, Kobe University, Kobe, Japan. [2]Department of Engineering Mathematics, University of Bristol, Bristol, UK. Correspondence and requests for materials should be addressed to N.M. (email: naoki.masuda@bristol.ac.uk)

their performance relative to the CI algorithm is unknown in general[5,10] or at least for networks with community structure[4,9]. Yet other community-based immunization algorithms impose that only local information about the network is available, mimicking realistic constraints[6,15,16]. This constraint naturally limits the performance of an immunization algorithm.

We develop an immunization algorithm by formulating a CI algorithm for a coarse-grained network, in which a node represents a community, and a weighted link represents the number of links between different communities. We compare the performance of the proposed algorithm with that of the CI algorithm[9], and the conventional algorithm targeting hubs[18–20], and others[5,10] when networks have community structure.

Theory

Consider an undirected and unweighted network having N nodes. The aim of an immunization algorithm is to sequentially remove nodes to fragment the network as soon as possible, i.e., with a small number of removed nodes.

Collective influence. The CI algorithm is based on the scoring of nodes according to the CI value[9]. The CI of node i is defined as

$$\mathrm{CI}_\ell(i) = z_i \sum_{j \in \partial \mathrm{Ball}(i,\ell)} z_j, \tag{1}$$

where

$$z_i \equiv k_i - 1, \tag{2}$$

k_i is the degree of node i, and $\partial\mathrm{Ball}\ (i,\ \ell)$ is the set of nodes at distance ℓ from node i. When $\ell = 0$, the CI is equivalent to the degree as long as the rank order is concerned.

The CI algorithm calculates the $\mathrm{CI}_\ell(i)$ value of all nodes and removes the node with the largest CI value in one step. Then, the CI values of all the remaining nodes are recalculated, and the same procedure is repeated.

In fact, we use the order of nodes to be removed determined above as a tentative order. To improve the overall performance, we reorder the nodes by reinserting them as follows. We start from the situation in which the fraction of nodes in the largest connected component (LCC) is equal to or less than 0.01 for the first time. Then, we calculate for each removed node i the number of components that node i connects if it is reinserted in the current network. Next, we add back the node that connects the smallest number of connected components. We repeat this procedure until all the removed nodes are reinserted such that the initial network is restored.

The computation time of the CI algorithm is evaluated as follows[9]. The calculation of $\mathrm{CI}_\ell(i)$ requires $O(1)$ time for one node, and hence $O(N)$ time for all nodes. Because sorting the $\mathrm{CI}_\ell(i)$ values consumes $O(N \log N)$ time, each step of the CI algorithm consumes $O(N \log N)$ time. Therefore, the total computation time until $O(N)$ nodes are removed is evaluated as $O(N^2 \log N)$. However, by exploiting the fact that the CI values of only $O(1)$ nodes are affected by the removal of a single node, one can accelerate the same algorithm with a max-heap data structure, yielding $O(N \log N)$ total computation time[25].

Community-based collective influence. Community structure may make a network not locally tree-like. We propose an immunization algorithm by running a weighted-network variant of the CI algorithm on a coarse-grained network in which a community constitutes a supernode. We first run a community detection algorithm. Denote by N_C the number of communities and by $\widetilde{A}$ the $N_\mathrm{C} \times N_\mathrm{C}$ coarse-grained weighted adjacency matrix whose (I, J) element is equal to the number of links that connect communities I and J ($I \neq J$). We use lowercases (e.g., i, j) to denote individual nodes and uppercases (e.g., I, J) to denote supernodes, i.e., communities, throughout the text. The diagonal elements of $\widetilde{A}$ are set to zero.

Assume that the coarse-grained network is locally tree-like. By taking into account the fact that the coarse-grained network is generally a weighted network, we define the CI of community I in the coarse-grained network by

$$\mathrm{CI}'_\ell(I) = z'_I \sum_{J \in \widetilde{\partial}\mathrm{Ball}(I,\ell)} z''_J, \tag{3}$$

where $\widetilde{\partial}\mathrm{Ball}\ (I,\ \ell)$ denotes the set of the communities whose distance from community I is equal to ℓ in the coarse-grained network.

We set

$$z'_I \equiv \sum_{I'=1}^{N_\mathrm{C}} \widetilde{A}_{II'} - \min_{I'} \widetilde{A}_{II'}. \tag{4}$$

This definition is analogous to $z_i \equiv k_i - 1$ in Eq. (1). With this definition of z'_I, the CI of community I is equal to zero when I has only one neighbor, as in the original CI[9,26].

We set

$$z''_J \equiv \sum_{J'=1}^{N_\mathrm{C}} \widetilde{A}_{JJ'} - \widetilde{A}_{JJ^-} \quad (\ell \geq 1), \tag{5}$$

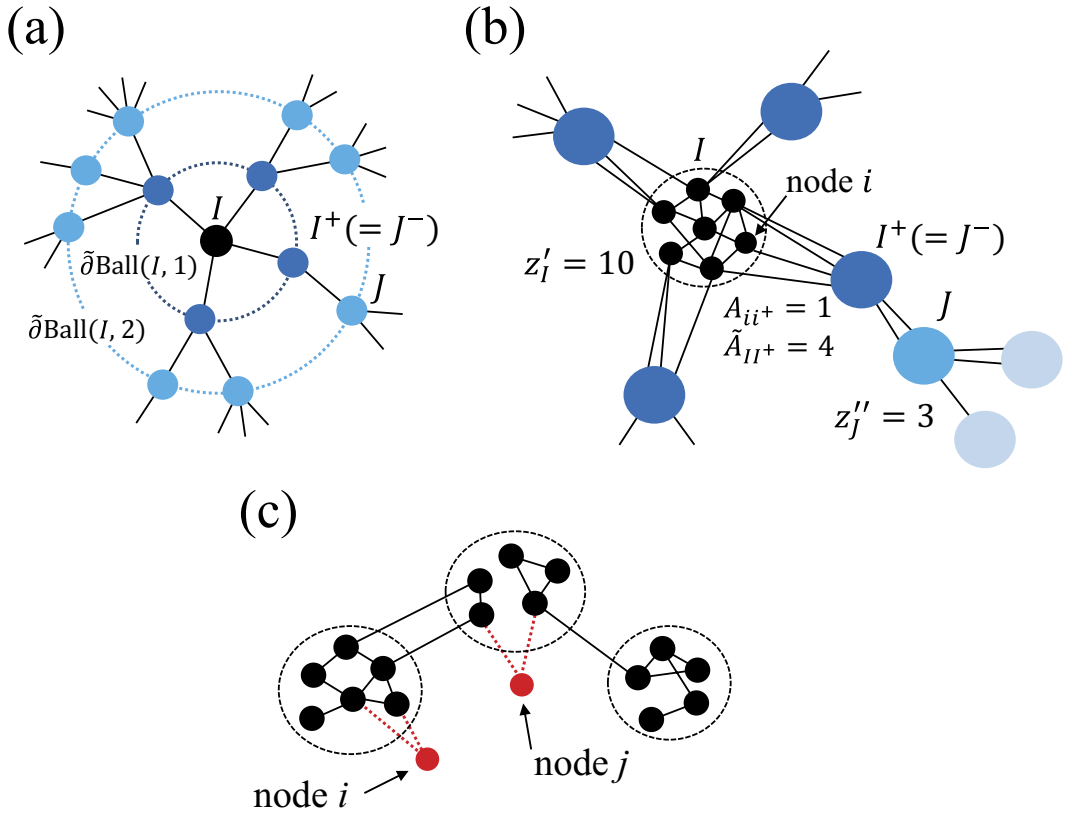

Figure 1. Concept of the community-based collective influence. (**a**) Egocentric view of the coarse-grained network. Each circle represents a community. Two communities are adjacent by a weighted link if a node in one community is connected to at least one node in the other community. The link weight in the coarse-grained network is equal to the number of links that connect the two communities in the original network. Local tree-like structure of the coarse-grained network is assumed. (**b**) Illustration of z'_I and z''_J for $\ell = 2$, in which case $I^+ = J^-$. A line represents a link in the original network. The dashed circle represents the Ith community. (**c**) Schematic of community-based reinsertion. A dashed circle represents a community. Suppose that we will reinsert either node i or j. If reinserted, node i and j would have a path to two and three communities, respectively. Therefore, we reinsert node i.

where J is a community that is at distance ℓ from I, and J^- is the community that is at distance $\ell - 1$ from I and on the path between I and J (Fig. 1(a)). It should be noted that z''_J is equal to zero if J^- is the only neighbor of J. It should also be noted that, when every community consists of only one node in the original network, $\mathrm{CI}'_\ell(i) = \mathrm{CI}_\ell(i)$ for $1 \le i \le N$. Equation (5) is ill-defined for $\ell = 0$. To be consistent with the original definition of the CI, we define $z''_J \equiv z'_J$ for $\ell = 0$. Then, $\mathrm{CI}'_0(I)$ is large when node I has a large degree in the coarse-grained network.

Let $A = (A_{ij})$ be the adjacency matrix of the original network. Equation (3) is rewritten as

$$\mathrm{CI}'_\ell(I) = z'_I \sum_{i \in \text{community} I} \sum_{I' \in \tilde{\partial}\mathrm{Ball}(I,1)} \frac{\sum_{i' \in I'} A_{ii'}}{\tilde{A}_{II'}} \sum_{\substack{J \in \tilde{\partial}\mathrm{Ball}(I,\ell) \\ I^+ = I'}} z''_J, \tag{6}$$

where I^+ is the community adjacent to I (hence distance one from I) through which J is reached from I (Fig. 1(b)). On the basis of Eq. (6), we define the community-based collective influence (CbCI) of node i, denoted by $\mathrm{CbCI}(i)$, as

$$\mathrm{CbCI}(i) = z'_I \sum_{I' \in \tilde{\partial}\mathrm{Ball}(I,1)} \frac{\sum_{i' \in I'} A_{ii'}}{\tilde{A}_{II'}} \sum_{\substack{J \in \tilde{\partial}\mathrm{Ball}(I,\ell) \\ I^+ = I'}} z''_J, \tag{7}$$

where node i belongs to community I. In Eq. (7), the importance of a node stems from three factors. First, $\mathrm{CbCI}(i)$ is proportional to z'_I, which is essentially the number of inter-community links of the community to which i belongs. Second, $\mathrm{CbCI}(i)$ is large if I has many high-degree nodes at distance ℓ in the coarse-grained network (i.e., sum of z''_J). Third, $\mathrm{CbCI}(i)$ is large if node i has many inter-community links relative to the total number of inter-community links that community I has (i.e., $\sum_{i' \in I'} A_{ii'} / \tilde{A}_{II'}$). We set $\ell = 2$ in the following numerical simulations. When $\ell = 2$, I^+ in Eqs (6) and (7) coincide with J^- in Eq. (5) (Fig. 1(b)).

We remove the node with the largest CbCI value. If there are multiple nodes with the same largest CbCI value, we select the node having the largest degree. If there are multiple nodes with the same largest CbCI and degree, we break the tie at random. Then, we recalculate the CbCI for all remaining nodes, remove the node with the largest CbCI, and repeat the same procedure until the size of the LCC becomes equal to or less than $0.01N$. We further optimize the obtained order of node removal by reinsertion, as in the CI algorithm. We use the coarse-grained network, not the original network, to inform the reinsertion process in the CbCI algorithm. In other words, the number of communities that belong to the same component as the reinserted node is measured for each tentatively reinserted node. We decide to reinsert the node whose presence connects the least number of communities (Fig. 1(c)).

Given a partitioning of the network into communities, the calculation of CbCI(i) for one node consumes $O(1)$ time. Therefore, if we adapt the original implementation of the CI algorithm[9] to the case of the CbCI, sorting of CbCI(i) dominates the computation time of the CbCI algorithm. The time complexity of the CbCI algorithm is the same as that of the CI algorithm in ref. 9, i.e., $O(N^2 \log N)$, if community detection is not a bottleneck. The use of the max-heap data structure makes the CbCI algorithm run in $O(N \log N)$ time if $N_C = O(N)$ such that the CbCI values of $O(1)$ nodes are affected by the removal of a single node. Generally speaking, the CbCI algorithm with the max-heap data structure runs in $O(N \log N) \times O(N/N_C) = O((N^2/N_C)\log N)$ time.

We use the following six algorithms for community detection: (i) Infomap[27,28], requiring $O(M)$ time[21], where M is the number of links, and hence $O(N)$ time for sparse networks; (ii) Walktrap, which requires $O(N^2 \log N)$ for most empirical networks[29]; (iii) the label-propagation algorithm, requiring nearly linear time in N ref. 30; (iv) a fast greedy algorithm for modularity maximization, requiring $O(N(\log N)^2)$ time for sparse networks[31]; (v) modularity maximization based on simulated annealing, which is practical up to $\approx 10^4$ nodes in the original paper[32] and time-consuming because modularity must be maximized in a parameter-dependent manner[33]; (vi) the Louvain algorithm, which practically runs in $O(N)$ time[34]. The last three algorithms intend to maximize the modularity, denoted by Q. The first three algorithms detect communities according to different criteria.

Except for the simulated annealing algorithm, the computational cost is at most that for the CbCI algorithm given the partitioning of the network, i.e., $O(N^2 \log N)$. Therefore, if the CbCI algorithm is naively implemented, community detection is not a bottleneck in terms of the computation time when any of these five community detection algorithms is used. If $N_C = O(N)$ and we implement the CbCI algorithm using the max-heap data structure, a community detection algorithm requiring more than $O(N \log N)$ time presents a bottleneck. In this case, the Infomap when the network is sparse (i.e., $M = O(N)$), label-propagation algorithm, and Louvain algorithm retain $O(N \log N)$ total computation time of the CbCI algorithm. The total computation time with any of the other three community detection algorithms is governed by that of the community detection algorithm.

Results

In this section, we compare the performance of the CbCI algorithm with the CI and other immunization algorithms (see Methods) on two model networks and 12 empirical networks. Let q be the fraction of removed nodes. The size of the LCC after qN nodes have been removed, divided by N, is denoted by $G(q)$.

Scale-free network models with and without community structure. We start by testing various immunization algorithms on a scale-free network model with built-in community structure (Methods). We sequentially remove nodes from this network according to each immunization algorithm and track the size of the LCC. We use the community structure imposed by the model to inform the CbCI and CbDI algorithms. The results for a range of immunization algorithms are shown in Fig. 2(a). Both CbCI and CbDI algorithms considerably outperform the CI algorithm. The CbCI algorithm performs better than the CbDI algorithm. The performance of the CbCI algorithm is close to the Betweenness algorithm. It should be noted that the Betweenness algorithm, while efficient, is not scalable to larger networks.

Next, we consider a scale-free network without community structure, which is generated by the original BA model with $N = 5000$ and $\langle k \rangle \approx 12$ (the parameters of the model are equal to $m_0 = m = 6$). We run the CbCI and CbDI strategies by applying a community detection algorithm to the generated network although the BA model lacks community structure. In fact, all but the label-propagation algorithm returns a partitioning result. The performance of the different immunization algorithms for this network is compared in Fig. 2(b). The CbCI algorithm combined with Infomap or Walktrap outperforms the Degree and LSP algorithms. The performance of the CbCI algorithm is close to that of the CI algorithm except in an early stage of node removal. A different community-based immunization algorithm, CbDI, lacks this feature. This result suggests that the CbCI algorithm combined with Infomap or Walktrap can work efficiently even when the network does not have community structure.

The results for the CbCI and CbDI algorithms combined with the other four community detection algorithms are shown in Fig. S2(a). The figure suggests that the CbCI algorithm combined with Infomap or Walktrap performs better than when it is combined with a different community detection algorithm.

Empirical networks. In this section, we run the CbCI and other algorithms on the following 12 empirical networks with community structure. (i) Two networks of Autonomous Systems of the Internet constructed by the University of Oregon Route Views project[35–37]: A node is an Autonomous System. The network collected on 2 January 2000 and that on 31 March 2001 are referred to as AS-1 and AS-2, respectively. (ii) Pretty Good Privacy network (PGP)[38]: Two persons are connected by a link if they share confidential information using the PGP encryption algorithm on the Internet. (iii) World Wide Web (WWW)[39]: A network of websites connected by hyperlinks, which is originally a directed network. (iv) Email-based communication network at Kiel University

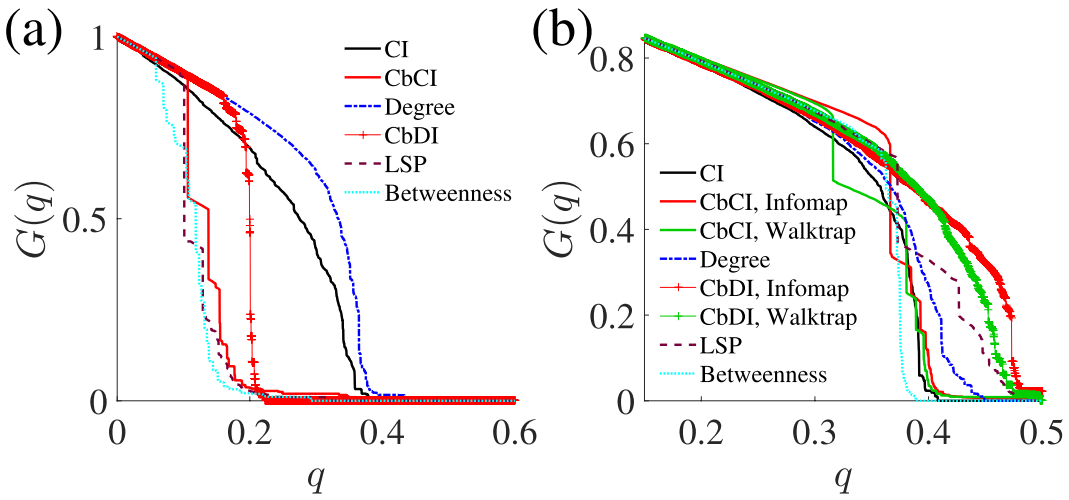

Figure 2. Normalized size of the LCC, $G(q)$, plotted against the fraction of removed nodes, q, in model networks with $N = 5000$. A curve corresponds to an immunization algorithm. See Methods for the abbreviations. (**a**) Scale-free network with prescribed community structure. (**b**) BA model.

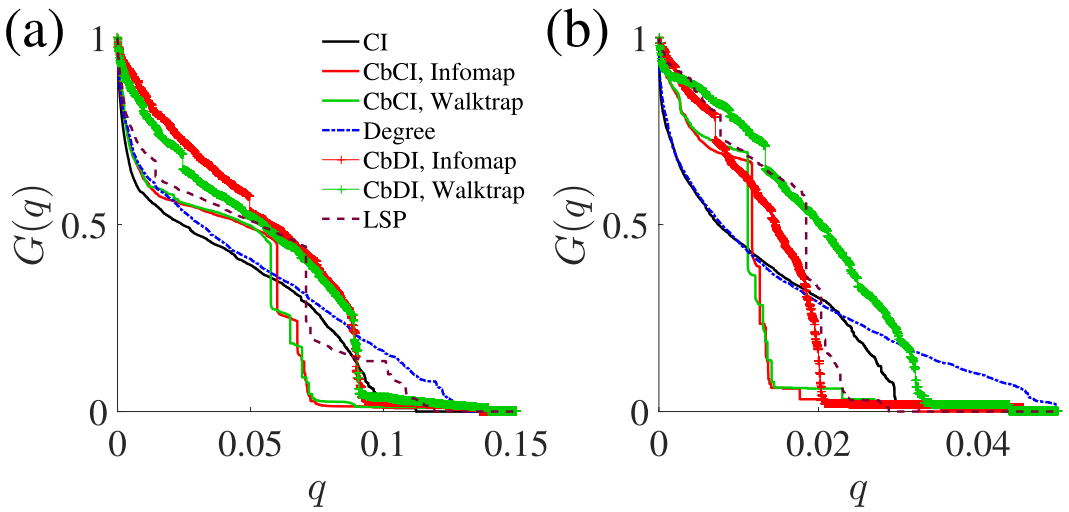

Figure 3. Normalized size of the LCC, $G(q)$, plotted against the fraction of removed nodes, q, in two empirical networks. (**a**) E-mail communication network in Enron. (**b**) World Wide Web.

(referred to as email-uni)[40]: E-mail sending activity among students, which provides a directed link, recorded over a period of 112 days. (v) Email-based communication network in Enron Corporation (email-Enron)[36,41,42]: Two e-mail users in the data set are connected by an unweighted directed link if at least one e-mail has been sent from one user to the other user. (vi) Collaboration networks in General Relativity and Quantum Cosmology (CA-GrQc), Astro Physics, (CA-Astroph), and Condensed Matter, (CA-Condmat) categories[36,43] and High Energy Physics – Phenomenology (CA-HepPh) and High Energy Physics – Theory (CA-HepTh) categories in arXiv[35,36]. By definition, two authors are adjacent if they coauthor a paper. (vii) High-energy physics citation network within the hep-th category of arXiv (HEP)[44], which is originally a directed network. For each network, we removed the link weight, self-loops, and direction of the link, and submitted the LCC to the following analysis. Summary statistics of these networks including the modularity, Q, are shown in Tables S1 and S2.

We do not investigate the Betweenness immunization algorithm due to its high computational cost (i.e., $O(NM)$ time for calculating the betweenness centrality of all nodes[24], hence $O(N^2M)$ time for removing $O(N)$ nodes).

The performance of the different immunization algorithms is compared on two empirical networks in Fig. 3. Among the 12 empirical networks that we tested, these two networks yielded the smallest and largest modularity values as maximized by the Louvain algorithm. The figure indicates that the CbCI algorithm combined with Infomap or Walktrap performs better than the previously proposed algorithms including the CI algorithm in both networks. The CbCI algorithm performs better than the CI algorithm in many other empirical networks as well

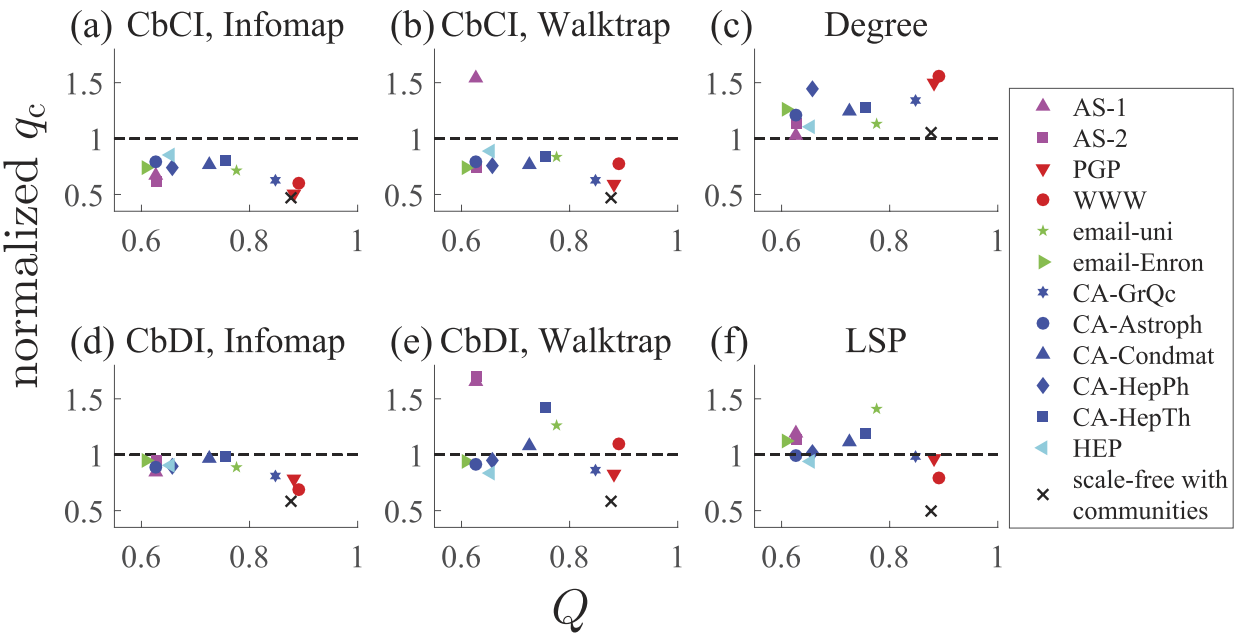

Figure 4. The fraction of removed nodes to fragment the network, q_c, for an immunization algorithm divided by the value for the CI algorithm. (**a**) CbCI combined with Infomap. (**b**) CbCI combined with Walktrap. (**c**) High degree adaptive (Degree). (**d**) CbDI combined with Infomap. (**e**) CbDI combined with Walktrap. (**f**) Laplacian spectral partitioning (LSP). A symbol represents a network. The cross represents the model network used in Fig. 2(a). The modularity value, Q, is determined by the Louvain algorithm.

(Fig. S2(b)–(m)). Furthermore, the CbCI algorithm combined with a different community detection algorithm also outperforms the CI algorithm in most of the networks (Fig. S2(b)–(m)).

To be quantitative, we measure the fraction of removed nodes at which the network fragments into sufficiently small connected components, i.e.,

$$q_c \equiv \inf\{q : G(q) < \theta\}, \tag{8}$$

where we remind that $G(q)$ is the size of the LCC normalized by N. We set $\theta = 0.05$. We calculate q_c for each combination of a network and an immunization algorithm.

The value of q_c for each immunization algorithm normalized by the q_c value for the CI algorithm is plotted in Fig. 4. A symbol represents a network. A small normalized value of q_c implies a high efficiency of the immunization algorithm. As expected, the Degree immunization algorithm performs worse than the CI in all the tested networks (Fig. 4(c)). For the CbCI algorithm combined with Infomap, q_c is smaller by 15.0% to 49.7% than that for the CI algorithm (Fig. 4(a)). The CbCI algorithm combined with Walktrap shows a similar performance for all but one networks (Fig. 4(b)). The CbCI algorithm combined with three of the other four community detection algorithms performs better than the CI algorithm for networks with relatively strong community structure (Fig. S3). The CbDI algorithm combined with Infomap performs better than the CI algorithm for all networks, but to a lesser extent than the CbCI algorithm combined with Infomap does (Fig. 4(d)). The CbDI algorithm combined with Walktrap (Fig. 4(e)) and the other four community detection algorithms (Fig. S3) performs worse than the CI algorithm. The LSP algorithm performs worse than the CI algorithm in a majority of the networks (Fig. 4(f)).

Even if two immunization algorithms yield the same q_c value on the same network, $G(q)$ may considerably drop at a smaller q value with one immunization algorithm than the other algorithm. To quantify the performance of immunization algorithms in this sense, we measure the size of the LCC integrated over q values[7,45], i.e.,

$$\overline{G} \equiv \frac{1}{N}\sum_{i=0}^{N} G(i/N). \tag{9}$$

It should be noted that $\overline{G}$ is the area under the curve when $G(q)$ is plotted against q and ranges between 0 and 1/2. A small $\overline{G}$ value implies a good performance of an immunization algorithm.

The value of $\overline{G}$ for each immunization algorithm normalized by that for the CI algorithm is plotted in Fig. 5. The CbCI algorithm combined with Infomap outperforms the CI algorithm in 11 out of the 12 networks in terms of $\overline{G}$ (Fig. 5(a)). Similarly, the CbCI algorithm combined with Walktrap outperforms the CI algorithm in ten out of the 12 networks (Fig. 5(b)). The CbCI combined with any of the other four community detection algorithms outperforms the CI algorithm in roughly half of the networks and tends to be efficient for networks having large modularity values as determined by the Louvain algorithm (Fig. S4). In particular, for the three networks with the largest modularity, the CbCI algorithm combined with any of the six community detection algorithms

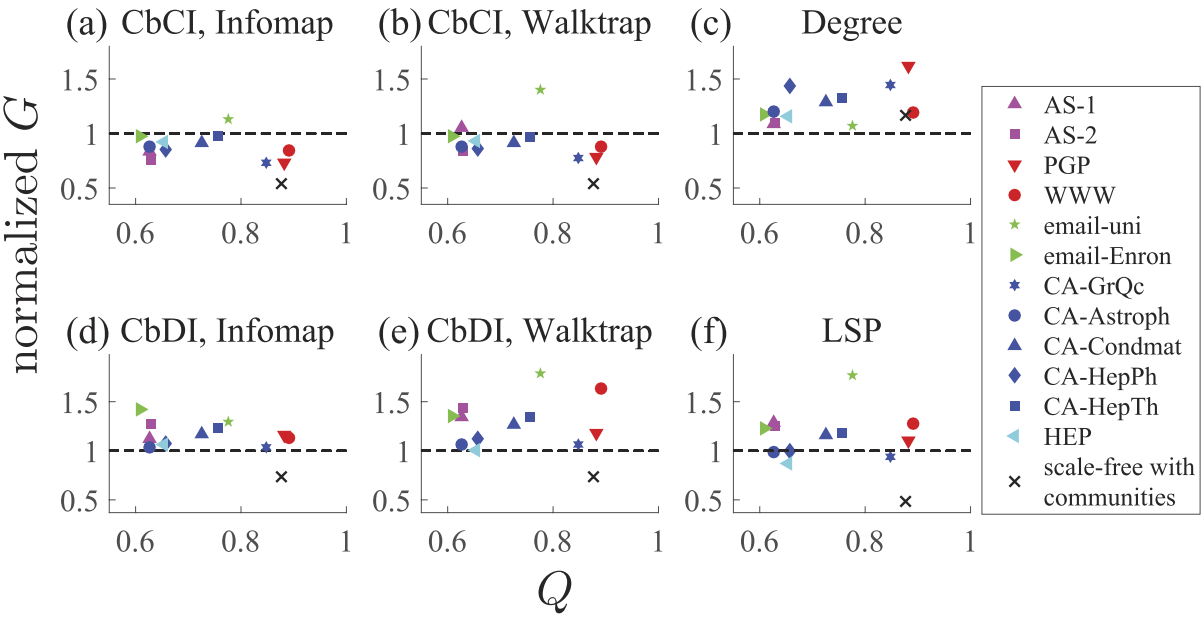

Figure 5. The $\overline{G}$ value normalized by that for the CI algorithm. (**a**) CbCI combined with Infomap. (**b**) CbCI combined with Walktrap. (**c**) Degree. (**d**) CbDI combined with Infomap. (**e**) CbDI combined with Walktrap. (**f**) LSP. The Q value is determined by the Louvain algorithm.

	Clustering coefficient	Weighted clustering coefficient	N_C	Mean path length	Entropy	Normalized entropy
AS-1	0.238	0.120	−0.415	−0.354	−0.523	−0.608
AS-2	−0.258	0.119	−0.417	0.065	−0.319	−0.203
PGP	0.298	0.490	−0.603	−0.534	−0.667	−0.781
WWW	−0.005	−0.430	0.306	−0.375	0.216	−0.169
email-uni	0.213	−0.053	−0.362	0.125	−0.446	−0.568
email-Enron	−0.278	−0.398	−0.073	−0.136	−0.650	−0.817
CA-GrQc	0.438	0.345	−0.773	−0.458	−0.891	−0.934
CA-Astroph	−0.154	−0.005	−0.406	−0.144	−0.764	−0.826
CA-Condmat	0.121	−0.181	−0.653	0.219	−0.820	−0.918
CA-HepPh	0.729	0.792	−0.845	−0.569	−0.932	−0.781
CA-HepTh	−0.200	−0.118	0.178	0.325	−0.067	−0.320
HEP	0.314	0.204	−0.718	−0.042	−0.842	−0.759

Table 1. Correlation coefficient between an explanatory variable and the normalized q_c. The clustering coefficient is defined by the number of triangles containing the ith node divided by $k_i(k_i-1)/2$, which is averaged over all nodes $1 \le i \le N$. The weighted clustering coefficient is defined by $\sum_{j,k=1;j,k\neq i}^{N}(\hat{w}_{ij}\hat{w}_{jk}\hat{w}_{ki})^{1/3}/[k_i(k_i-1)]$, which is averaged over i[48,49]. Here, $\hat{w}_{ij} = w_{ij}/\max_{1\le i',j'\le N}(w_{i'j'})$, and w_{ij} is the weight of the link between the ith and jth nodes. We calculated the correlation coefficient for each network on the basis of the data points obtained from the six community detection algorithms. The scattergrams based on which the correlation coefficient has been calculated are shown in Fig. S5(a), (c), (e), (g), (i), and (k).

outperforms the CI algorithm. The Degree, CbDI, and LSP algorithms are less efficient than the CI algorithm in terms of $\overline{G}$ (Figs 5(c)–(f) and S4).

Why do Infomap and Walktrap marry better with the CbCI algorithm than the other community detection algorithms? We have shown that the CbCI algorithm is more efficient when it is combined with Infomap or Walktrap, in particular Infomap, than with the other four community detection algorithms. To explore why, we start by measuring the clustering coefficient[46] of the unweighted version of the coarse-grained networks. We do so because in theory the CI assumes locally tree-like networks[9,47]. High clustering in the coarse-grained network may discourage the CbCI algorithm. For each empirical network, we measure the Pearson correlation coefficient between the clustering coefficient and q_c normalized by the value for the CI algorithm. We use the result for each community detection algorithm as a data point such that the correlation coefficient is calculated on the basis of six data points. The results are shown in Table 1. We find that the clustering coefficient is not consistently correlated with the normalized q_c. The results are qualitatively the same with a

	Clustering coefficient	Weighted clustering coefficient	N_C	Mean path length	Entropy	Normalized entropy
AS-1	0.064	0.284	−0.813	−0.259	−0.813	−0.604
AS-2	−0.369	0.416	−0.693	0.105	−0.660	−0.449
PGP	0.084	0.295	−0.489	−0.586	−0.603	−0.766
WWW	−0.367	0.111	−0.713	0.281	−0.820	−0.694
email-uni	0.394	0.362	−0.810	0.043	−0.823	−0.595
email-Enron	−0.381	−0.224	−0.474	−0.258	−0.804	−0.798
CA-GrQc	0.034	0.045	−0.467	−0.043	−0.679	−0.909
CA-Astroph	−0.313	−0.112	−0.137	−0.232	−0.526	−0.700
CA-Condmat	0.334	−0.024	−0.706	0.387	−0.851	−0.839
CA-HepPh	0.227	0.419	−0.424	−0.214	−0.607	−0.685
CA-HepTh	0.248	0.629	−0.632	−0.129	−0.754	−0.730
HEP	−0.067	0.400	−0.636	0.395	−0.759	−0.722

Table 2. Correlation coefficient between an explanatory variable and the normalized $\overline{G}$ for each network. We calculate the correlation coefficient for each network on the basis of the data points obtained from the six community detection algorithms. The scattergrams based on which the correlation coefficient has been calculated are shown in Fig. S5(b), (d), (f), (h), (j), and (l).

weighted clustering coefficient[48,49] (Table 1). We obtain similar results if $\overline{G}$ instead of q_c is used as a performance measure (Table 2). It should be noted that different community detection algorithms yield sufficiently different clustering coefficient values including large values (Fig. S5(a)). We conclude that the lack of local tree-like structure in the coarse-grained networks is not a strong determinant of the performance of the CbCI algorithm. This result does not contradict those for the original CI algorithm, which assumes local tree-like networks, because the CI algorithm is practically efficient on loopy networks as well[9].

We have set $\ell = 2$, thus ignoring the contribution of nodes in coarse-grained networks three or more hops away from a focal node. In fact, large coarse-grained networks may have a large mean path length and deteriorate the performance of the CbCI algorithm. Therefore, we calculate the correlation coefficient between N_C, i.e., the number of the detected communities, and q_C, and between the mean path length in the unweighted coarse-grained network and q_C (Table 1). The correlation efficient between $\overline{G}$ and either N_C or the mean path length is also measured (Table 2). The tables indicate that the performance of a community detection algorithm is not consistently correlated with the mean path length. It is correlated with N_C, but in the manner such that the performance of the CbCI algorithm improves as N_C increases, contrary to the aforementioned postulated mechanism. Therefore, the use of $\ell = 2$ does not probably explain the reason why a community detection algorithm marries the CbCI algorithm better than another.

In fact, the CbCI algorithm performs well when the detected communities have relatively similar sizes. To show this, we measure the entropy in the partitioning, which is defined by $\sum_{c=1}^{N_C}(N_c'/N)\log(N_c'/N)$, where N_c' is the number of nodes in the cth community. The entropy ranges between 0 and $\log N_C$. A large entropy value implies that the partitioning of the network is relatively egalitarian. The correlation coefficient between the entropy and the normalized q_c is shown in Table 1 for each network. The entropy and q_c are negatively correlated with each other for all networks and strongly so for most of the networks. This result is robust when we normalize the entropy by the largest possible value, i.e., $\log N_C$ (Table 1), and when the performance measure is replaced by $\overline{G}$ (Table 2).

To assess the robustness of this finding, we calculate the same correlation coefficient between either the unnormalized or normalized entropy and one of the two performance measures, but for each community detection algorithm. Now each empirical network constitutes a data point based on which the correlation coefficient is calculated. The correlation coefficient values are shown in Table 3. Although the correlation is weaker than in the previous case, the correlation between the entropy and either the normalized q_C or $\overline{G}$ is largely negative, which is consistent with the results shown in Tables 1 and 2. The correlation coefficient between Q and each of the performance measure is also shown in Table 3. The entropy provides a weaker determinant of the performance as compared to Q, which is expected because the CbCI algorithm is designed for networks with community structure. Nevertheless, the entropy provides a larger (i.e., more negative) correlation value than Q does in some cases (Table 3).

Infomap tends to detect a large number of communities (Table S2) whose size is less heterogeneously distributed than the case of the other community detection algorithms (Fig. S5(i) and (k)). We consider that this is a main reason why Infomap is effective when combined with the CbCI algorithm. Roughly speaking, the label-propagation algorithm tends to yield a similarly large number of communities, N_C (Table S2). However, the size of the community is more heterogeneously distributed with the label-propagation algorithm than with Infomap, as quantified by the entropy measures (Fig. S5(i) and (k)).

Discussion

We showed that the CbCI immunization algorithm outperforms the CI and some other algorithms when a given network has community structure. The algorithm aims to pinpoint nodes that connect different communities at a reasonable computational cost. The CbCI algorithm is in particular efficient when Infomap[27,28] is used

	Entropy		Normalized entropy		Q	
	q_c	$\overline{G}$	q_c	$\overline{G}$	q_c	$\overline{G}$
Infomap	−0.088	0.358	−0.017	−0.206	−0.576	−0.210
Walktrap	−0.506	0.014	−0.434	−0.201	−0.348	−0.128
label propagation	−0.690	−0.659	−0.630	−0.687	−0.689	−0.797
fast greedy	−0.211	−0.099	0.067	−0.081	−0.015	−0.310
simulated annealing	−0.695	0.057	−0.288	−0.005	−0.330	−0.391
Louvain	−0.791	−0.001	−0.267	−0.370	−0.707	−0.370

Table 3. Correlation coefficient between an explanatory variable and a performance measure for each community detection algorithm. We calculate the correlation coefficient for each community detection algorithm on the basis of the data points obtained from the 12 empirical networks. q_c and $\overline{G}$ indicate the values normalized by those for the CI algorithm. The scattergrams based on which the correlation coefficient has been calculated are shown in Fig. S6.

for detecting communities beforehand. Infomap runs sufficiently fast at least for sparse networks[21] such that the entire CbCI algorithm runs as fast as the CI algorithm at least asymptotically in terms of the network size. The Walktrap community detection algorithm[29] is the second best among the six candidates to be combined with the CbCI algorithm in terms of the quality of immunization. However, Walktrap is slower than Infomap. Walktrap consumes longer time than the main part of the CbCI algorithm, i.e., sequential node removal, when the max-heap data structure is used for implementing the CbCI algorithm. In this case, the community detection before starting the node removal is the bottleneck of the entire CbCI algorithm, and the CbCI algorithm is slower than the CI algorithm. To our numerical efforts, we recommend Infomap to be combined with the CbCI algorithm.

We argued that Infomap works better in combination with the CbCI algorithm than the other community detection algorithms do mainly because Infomap yields a relatively egalitarian distribution of the community size. However, the distribution of the community size is usually skewed even with Infomap[50]. The CbCI algorithm may work even better if we use a community detection algorithm that imposes that the detected communities are of the equal or similar sizes. This problem is known as k-balanced partitioning, where k refers to the number of communities. Although k-balanced partitioning for general k is notoriously hard to solve, there are various approximate algorithms for this problem[51–53]. Combining these algorithms with the CbCI algorithm may be profitable.

We partitioned the network just once in the beginning of the CbCI algorithm and used the obtained community structure throughout the node removal procedure. This property is shared by the CbDI algorithm[5] and another immunization algorithm[11]. We may be able to improve the performance of immunization by updating the community structure during the node removal. Our preliminary numerical simulations did not yield an improvement of the CbCI algorithm with online updating of community structure (section S1 in the SI). We should also bear in mind the computational cost of community detection, which would be repeatedly applied in the case of online updating. Nevertheless, this line of improvement may be worth investigating.

The CI assumes locally tree-like networks[9]. Although the CI algorithm is practically efficient in moderately loopy networks as well[9], many empirical networks are abundant in triangles and short cycles such that they are highly loopy[1]. Dense connectivity within a community implies that there tend to be many triangles and short cycles in a network with community structure[54,55]. Then, coarse graining effectively coalesces many triangles and short cycles into one supernode, possibly suppressing their detrimental effects. At the same time, however, coarse-grained networks tend to have a large clustering coefficient (Fig. S5(a)). We may be able to improve the performance of the CbCI algorithm by suppressing the effect of short cycles in coarse-grained networks. Recently, a method has been proposed to improve the accuracy of estimating the percolation threshold using non-backtracking matrices, where redundant paths are suppressed in the counting of the paths[47]. This method applied to both CI and CbCI algorithms may enhance their performance in the immunization problem.

The recently proposed collective influence propagation (CI_p) algorithm, which can be interpreted as the CI algorithm in the limit of $\ell \to \infty$, generally yields better solutions than the CI algorithm does[25]. Given that we have not implemented the CI_p algorithm in the present article, we are not arguing that the CbCI algorithm is better than the CI_p algorithm. It should also be noted that we may be able to combine the CbCI algorithm with the idea of the CI_p algorithm (i.e., using the leading left and right eigenvectors of the non-backtracking matrix) to devise a new algorithm.

Methods

Immunization algorithms to be compared. We compare the performance of the CI and CbCI algorithms against the following immunization algorithms.

- High degree adaptive (abbreviated as Degree)[18–20]: We sequentially remove the node having the largest degree. If multiple nodes have the largest degree, we break the tie by selecting one of the largest-degree nodes at random. We recalculate the degree after each node has been removed.

- Community-based dynamical importance (CbDI)[5]: This method exploits the community structure of a network, similar to the CbCI algorithm, but calculates the importance of a community in the coarse-grained

network in terms of the so-called dynamical importance[3]. The CbDI algorithm needs a community detection algorithm. We use each of the six community detection algorithms used in the CbCI algorithm.

The CbDI algorithm runs as follows[5]. We denote by $\widetilde{\lambda}$ and $\widetilde{\boldsymbol{u}} = (\widetilde{u}_1 \cdots \widetilde{u}_{N_C})^T$ the largest eigenvalue and the corresponding eigenvector of $\widetilde{A}$, respectively. Owing to the Perron-Frobenius theorem, it holds true that $\widetilde{\lambda} > 0$ and $\widetilde{u}_i \geq 0$ $(1 \leq i \leq N_C)$. The number of links between node i and the Jth community is denoted by $k_{iJ} \equiv \sum_{j \in \text{community}J} A_{ij}$. We define $x = \left(\sum_{J=1, J\neq I}^{N_C} k_{ij}\widetilde{u}_J\right)/\widetilde{\lambda}$, where I is the community to which node i belongs. The CbDI of node i is defined by $(2\widetilde{u}_I - x)\sum_{J=1, J\neq I}^{N_C} k_{ij}\widetilde{u}_J$. We remove the nodes in descending order of the CbDI. If there are multiple nodes that have the same largest CbDI value, we break the tie by selecting the node that has the largest number of intra-community links. We recalculate the CbDI values of all the remaining nodes after removing each node. Once all the communities are disconnected, we sequentially remove the nodes in descending order of k_{iI}. We recalculate k_{iI} of all the remaining nodes after removing each node.

- The Laplacian spectral partitioning (LSP) algorithm runs as follows[10]:
  1. For the largest connected component (LCC), calculate the Fiedler vector, i.e., the eigenvector associated with the smallest positive eigenvalue of the Laplacian, $L \equiv D_{LCC} - A_{LCC}$, where D_{LCC} denotes the $N_{LCC} \times N_{LCC}$ diagonal matrix whose (i, i) element is equal to the degree of the ith node in the LCC, N_{LCC} is the number of nodes in the LCC, and A_{LCC} is the adjacency matrix of the LCC.
  2. Partition the N_{LCC} nodes into two non-empty groups by thresholding on the value of the element in the Fiedler vector. Group 1 (group 2) consists of the nodes whose corresponding element in the Fiedler vector is higher (lower) than a threshold. There are $N_{LCC} - 1$ possible ways to bipartition the nodes.
  3. Calculate

$$\mathcal{Q} = m_{in} \ln\left(\frac{2m_{in}}{K_1^2 + K_2^2}\right) + m_{out} \ln\left(\frac{m_{out}}{K_1 K_2}\right), \tag{10}$$

  for each bipartition, where m_{in} and m_{out} are the numbers of intra-group and inter-group links, respectively. K_1 and K_2 represent the sum of the nodes' degrees in groups 1 and 2, respectively.

  4. Find the partition that maximizes $\mathcal{Q}$.
  5. Given the partition, remove the node that has the largest number of inter-group links. Then, recalculate the number of inter-group links for each remaining node. Repeat the node removal until the two groups are disconnected.
  6. Repeat steps 1–5 until the size of the LCC becomes less than θN, where $\theta = 0.01$.

- High betweenness centrality adaptive (abbreviated as Betweenness)[6,16,22,23]: We remove the node with the largest betweenness centrality. If multiple nodes have the same largest betweenness centrality value, the node having the largest degree is removed. We recalculate the betweenness of all nodes every time we remove a node.

We excluded the dynamical importance[3] because it is less successful than the CI on various networks[9] and than the CbDI on networks with community structure[5]. We also excluded the immunization algorithms on the basis of the PageRank, closeness centrality, and k-core, which had been shown to be outperformed by the CI algorithm[9]. This is because these algorithms do not particularly exploit community structure of the network such that there is no reason for believing that they would perform competitively on networks with community structure.

A scale-free network model with community structure. We constructed a scale-free network with built-in community structure as follows[5]. We first generate a coarse-grained network whose node is regarded as community, using the Barabási-Albert (BA) model[56] having $N_C = 100$ nodes and mean degree six. The initial network is the clique composed of $m_0 = 3$ nodes, and each added node has $m = 3$ links. After generating a coarse-grained network, we assign 50 nodes to each community, resulting in $N = 50 \times N_C = 5000$ nodes in total. For each community, the intra-community network is given by the BA model with $m_0 = m = 4$, which yields the mean within-community degree equal to $\langle k \rangle_\ell = 2[(N - m_0)m + m_0(m_0 - 1)/2]/N = 7.6$. Additionally, if communities I and J are adjacent in the coarse-grained network, then nodes $i \in I$ and $j \in J$ are connected with probability $\langle k \rangle_g/(6N/N_C)$. This guarantees that a node is adjacent to $\langle k \rangle_g$ nodes in different communities on average. We set $\langle k \rangle_g = 1$. The mean degree of the entire network is equal to $\langle k \rangle = 8.58 \approx \langle k \rangle_\ell + \langle k \rangle_g$.

References

1. Newman, M. E. J. *Networks — An Introduction*. Oxford University Press, Oxford (2010).
2. Altarelli, F., Braunstein, A., Dall'Asta, L., Wakeling, J. R. & Zecchina, R. Containing epidemic outbreaks by message-passing techniques. *Phys. Rev. X* **4,** 021024 (2014).
3. Restrepo, J. G., Ott, E. & Hunt, B. R. Weighted percolation on directed networks. *Phys. Rev. Lett.* **100,** 058701 (2008).
4. Chen, Y., Paul, G., Havlin, S., Liljeros, F. & Stanley, H. E. Finding a better immunization strategy. *Phys. Rev. Lett.* **101,** 058701 (2008).
5. Masuda, N. Immunization of networks with community structure. *New J. Phys.* **11,** 123018 (2009).
6. Salathé, M. & Jones, J. H. Dynamics and control of diseases in networks with community structure. *PLOS Comput. Biol.* **6,** e1000736 (2010).
7. Schneider, C. M., Mihaljev, T., Havlin, S. & Herrmann, H. J. Suppressing epidemics with a limited amount of immunization units. *Phys. Rev. E* **84,** 061911 (2011).
8. Zhao, D. *et al.* Immunization of epidemics in multiplex networks. *PLOS ONE* **9,** e112018 (2014).
9. Morone, F. & Makse, H. A. Influence maximization in complex networks through optimal percolation. *Nature* **524,** 65–68 (2015).

10. Zahedi, R. & Khansari, M. A new immunization algorithm based on spectral properties for complex networks. In *The 7th Conference on Information and Knowledge Technology (IKT)*, 1–5 (2015).
11. Requião da Cunha, B., González-Avella, J. C. & Gonçalves, S. Fast fragmentation of networks using module-based attacks. *PLOS ONE* **10,** e0142824 (2015).
12. Mugisha, S. & Zhou, H.-J. Identifying optimal targets of network attack by belief propagation. *Phys. Rev. E* **94,** 012305 (2016).
13. Cohen, R., Havlin, S. & ben-Avraham, D. Efficient immunization strategies for computer networks and populations. *Phys. Rev. Lett.* **91,** 247901 (2003).
14. Gallos, L. K., Liljeros, F., Argyrakis, P., Bunde, A. & Havlin, S. Improving immunization strategies. *Phys. Rev. E* **75,** 045104(R) (2007).
15. Gong, K. *et al.* An efficient immunization strategy for community networks. *PLOS ONE* **8,** e83489 (2013).
16. Hébert-Dufresne, L., Allard, A., Young, J.-G. & Dubé, L. J. Global efficiency of local immunization on complex networks. *Sci. Rep*. **3,** 2171 (2013).
17. Liu, Y., Deng, Y. & Wei, B. Local immunization strategy based on the scores of nodes. *Chaos* **26,** 013106 (2016).
18. Albert, R., Jeong, H. & Barabási, A.-L. Error and attack tolerance of complex networks. *Nature* **406,** 378–382 (2000).
19. Callaway, D. S., Newman, M. E. J., Strogatz, S. H. & Watts, D. J. Network robustness and fragility: Percolation on random graphs. *Phys. Rev. Lett*. **85,** 5468–5471 (2000).
20. Cohen, R., Erez, K., ben-Avraham, D. & Havlin, S. Breakdown of the internet under intentional attack. *Phys. Rev. Lett*. **86,** 3682–3685 (2001).
21. Fortunato, S. Community detection in graphs. *Phys. Rep.* **486,** 75–174 (2010).
22. Holme, P., Kim, B. J., Yoon, C. N. & Han, S. K. Attack vulnerability of complex networks. *Phys Rev. E* **65,** 056109 (2002).
23. Ueno, T. & Masuda, N. Controlling nosocomial infection based on structure of hospital social networks. *J. Theor. Biol.* **254,** 655–666 (2008).
24. Brandes, U. A faster algorithm for betweenness centrality. *J. Math. Sociol.* **25,** 163–177 (2001).
25. Morone, F., Min, B., Bo, L., Mari, R. & Makse, H. A. Collective influence algorithm to find influencers via optimal percolation in massively large social media. *Sci. Rep*. **6,** 30062 (2016).
26. Karrer, B., Newman, M. E. J. & Zdeborová, L. Percolation on sparse networks. *Phys. Rev. Lett*. **113,** 208702 (2014).
27. Rosvall, M. & Bergstrom, C. T. Maps of random walks on complex networks reveal community structure. *Proc. Natl. Acad. Sci. USA* **105,** 1118–1123 (2008).
28. Rosvall, M., Axelsson, D. & Bergstrom, C. T. The map equation. *Eur. Phys. J. Spec. Top.* **178,** 13–23 (2010).
29. Pons, P. & Latapy, M. Computing communities in large networks using random walks. In *Computer and Information Sciences-ISCIS 2005*, Yolum, P., Güngör, T., Gürgen, F. & Özturan, C. editors, volume 3733 of *Lecture Notes in Computer Science*, 284–293. Springer Berlin Heidelberg (2005).
30. Raghavan, U. N., Albert, R. & Kumara, S. Near linear time algorithm to detect community structures in large-scale networks. *Phys. Rev. E* **76,** 036106 (2007).
31. Clauset, A., Newman, M. E. J. & Moore, C. Finding community structure in very large networks. *Phys. Rev. E* **70,** 066111 (2004).
32. Sales-Pardo, M., Guimerá, R., Moreira, A. A. & Amaral, L. A. N. Extracting the hierarchical organization of complex systems. *Proc. Natl. Acad. Sci. USA* **104,** 15224–15229 (2007).
33. Lancichinetti, A. & Fortunato, S. Community detection algorithms: A comparative analysis. *Phys. Rev. E* **80,** 056117 (2009).
34. Blondel, V. D., Guillaume, J.-L., Lambiotte, R. & Lefebvre, E. Fast unfolding of communities in large networks. *J. Stat. Mech*. **2008,** P10008 (2008).
35. Leskovec, J., Kleinberg, J. & Faloutsos, C. Graphs over time: densification laws, shrinking diameters and possible explanations. In *Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining*, 177–187. ACM (2005).
36. Jure, L. & Andrej, K. Stanford Network Analysis Project. http://snap.stanford.edu/. (Date of access: 19/01/2016).
37. University of Oregon Route Views Project. http://www.routeviews.org. (Date of access: 19/01/2016).
38. Boguñá, M., Pastor-Satorras, R., Díaz-Guilera, A. & Arenas, A. Models of social networks based on social distance attachment. *Phys. Rev. E* **70,** 056122 (2004).
39. Albert, R., Jeong, H. & Barabási, A.-L. Internet: Diameter of the world-wide web. *Nature* **401,** 130–131 (1999).
40. Ebel, H., Mielsch, L. I. & Bornholdt, S. Scale-free topology of e-mail networks. *Phys. Rev. E* **66,** 035103(R) (2002).
41. Klimt, B. & Yang, Y. The enron corpus: A new dataset for email classification research. In *Machine Learning: ECML 2004*, 217–226, Springer (2004).
42. Leskovec, J., Lang, K. J., Dasgupta, A. & Mahoney, M. W. Community structure in large networks: Natural cluster sizes and the absence of large well-defined clusters. *Internet Math.* **6,** 29–123 (2009).
43. Leskovec, J., Kleinberg, J. & Faloutsos, C. Graph evolution: Densification and shrinking diameters. *ACM Trans. Knowl. Discov. Data* **1,** 2 (2007).
44. Batagelj, V. & Mrvar, A. Pajeck datasets (2006), http://vlado.fmf.uni-lj.si/pub/networks/data/. (Data of access: 19/01/2016).
45. Schneider, C. M., Moreira, A. A., Andrade, J. S., Havlin, S. & Herrmann, H. J. Mitigation of malicious attacks on networks. *Proc. Natl. Acad. Sci. USA* **108,** 3838–3841 (2011).
46. Watts, D. J. & Strogatz, S. H. Collective dynamics of 'small-world' networks. *Nature* **393,** 440–442 (1998).
47. Radicchi, F. & Castellano, C. Beyond the locally treelike approximation for percolation on real networks. *Phys. Rev. E* **93,** 030302(R) (2016).
48. Onnela, J.-P., Saramäki, J., Kertész, J. & Kaski, K. Intensity and coherence of motifs in weighted complex networks. *Phys. Rev. E* **71,** 065103 (2005).
49. Saramäki, J., Kivelä, M., Onnela, J.-P., Kaski, K. & Kertész, J. Generalizations of the clustering coefficient to weighted complex networks. *Phys. Rev. E* **75,** 027105 (2007).
50. Lancichinetti, A., Kivelä, M., Saramäki, J. & Fortunato, S. Characterizing the community structure of complex networks. *PLOS ONE* **5,** e11976 (2010).
51. Andreev, K. & Racke, H. Balanced graph partitioning. *Theor. Comp. Sys.* **39,** 929–939 (2006).
52. Krauthgamer, R., Naor, J. S. & Schwartz, R. Partitioning graphs into balanced components. In *Proceedings of the Twentieth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '09, 942–949 (2009).
53. Feldmann, A. E. & Foschini, L. Balanced partitions of trees and applications. *Algorithmica* **71,** 354–376 (2015).
54. Radicchi, F., Castellano, C., Cecconi, F., Loreto, V. & Parisi, D. Defining and identifying communities in networks. *Proc. Natl. Acad. Sci. USA* **101,** 2658–2663 (2004).
55. Palla, G., Derényi, I., Farkas, I. & Vicsek, T. Uncovering the overlapping community structure of complex networks in nature and society. *Nature* **435,** 814–818 (2005).
56. Barabási, A.-L. & Albert, R. Emergence of scaling in random networks. *Science* **286,** 509–512 (1999).

Acknowledgements

N.M. acknowledges the support provided through JST, CREST, and JST, ERATO, Kawarabayashi Large Graph Project. T.K. acknowledges financial support from the Japan Society for the Promotion of Science KAKENHI Grants no. 25780203, 15H01948, and 16K03551. We thank Flaviano Morone and Taro Takaguchi for providing codes for the CI algorithm.

Author Contributions

T.K. and N.M. conceived the research. T.K. conducted the analysis. T.K. and N.M. discussed the results and wrote the manuscript.

Additional Information

Supplementary information accompanies this paper at http://www.nature.com/srep

Competing financial interests: The authors declare no competing financial interests.

How to cite this article: Kobayashi, T. and Masuda, N. Fragmenting networks by targeting collective influencers at a mesoscopic level. *Sci. Rep.* **6**, 37778; doi: 10.1038/srep37778 (2016).

Publisher's note: Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.